International Journal of Engineering & Technology, 7 (2.3) (2018) 1-3

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research Paper

Towards a new framework of program quality measurement based on programming language standards

Mohammad M. A. Abdallah*, Mustafa M. Alrifaee

Al-Zaytoonah University of Jordan, Amman, Jordan
**Corresponding author E-mail: m.abdallah@zuj.edu.jo*

Abstract

All languages, natural and programming, have rules and styles in how to write. These rules and styles mainly aim to make sure that anyone, who understand a language, can understand what the sentence say. In other words, the aim of rules and styles in a language is to deliver an information to reader, and the reader must get the right information. The literature review shows a lack of studies focusing on the code writing standards measurement processes. In this paper, we proposed a framework that can be applied on any programming language, using any standard of that language.

Keywords: *Standards, Programming language, programming style, programming measurement.*

1. Introduction

Ferris has introduced a definition of software measurement [1] as "Measurement is an empirical process, using an instrument, effecting a rigorous and objective mapping of an observable into a category in a model of the observable that meaningfully distinguishes the manifestation from other possible and distinguishable manifestations". This definition allows us to identify such objects according to defined rules. In software development, measurements are performed by using metrics, which are experimental designations of a value to an object aiming to characterize a definite quality of this object.

The goal of software measurement is to supply useful information not just data. Many organizations have recognized the need to improve their software development processes and to make their technical departments more cost effective. However, without sufficient information to indicate the problems areas they spend considerable amounts of money on new case tools, development techniques and hardware and software technologies without confirmed evidence that they will address the problems. Measurement provides the information for management to make informed decisions on managing IT resources [2, 3].

Moreover, the quality of code writing has the smallest share of the software measurement studies, where is few studies that discuss it as will be seen in section 2. In section 3, the proposed framework is explained, and is evaluated in section 4. Future work and conclusions are in section 5.

pages.

2. Related work

Measures are usually used to assess the quality of a product or a system. In Software development cycle, measures help in better understanding then improve the model that has been used in continuous basis. Moreover, Measurement helps in estimation, quality control, productivity assessment and project control throughout a software project, leading to strategic decision-making as project proceeds.

Regarding to [4] the software metrics can be classified into four types:

1. Based on process and product: where these metrics are measuring the software product in any stage of its development process. Such as requirement [5, 6], design [7], validation and verification [8-10].
2. Based on the metrics conditions on results, where this kind of metrics is focused on the objectives and subjective of the software metrics. In these metrics, the software objectives are quite clear to be identified and measured, however, the subjective are still not clear to be identified and can be different from one quality assurance team to another [11-13].
3. Based on computation, where the metrics can be categorized based on computation as "primitive metrics or computed metrics" [14]. And "Primitive software metrics" are those that can be directly observed. "Computed software metrics" are those that cannot be directly observed but are computed in some manner from other software metrics like in [15-17].
4. Metrics based on software development, the software development metrics can be classified based on software development model as Procedural Metrics (PM) [18, 19], Object-Oriented Metrics (OOM) [20, 21], and Web Metrics (WM), is to defined the success of a website [22].

In addition, there are different software metrics that not mentioned above, such as the security metrics, performance metrics, and many other metrics that can be found in [23].

In the previous work, the researchers focused on reliability, dependability, security, project management factors to measure the software quality. On the other side, only few researchers and companies are use program language standards to measure the software quality.

Abdallah [24, 25] used MISRA C standards [26] are used to measure the programs that written in C language. Program Clause slicing [24, 27] was used to analyse the code and check it against the standards. LDRA [28] and FlexeLint [29] has measured pro-

grams using the MISRA C language standards in addition to their own Standards.
ATHENA [30] is a software tool oriented towards the measurement of software quality characteristics. Measurement is based on the syntax and semantic (graph-based) analysis of programs.
The Consortium for IT Software Quality (CISQ) has introduced standards that are now "acquisition-ready" for managing system "utilities" such as security, reliability, performance efficiency, and maintainability – from system source code. In October 2017, CISQ also introduced a new OMG standard for measuring technical debt, which is a useful metric in the IT modernization and security discussion [31].

3. The proposed quality framework

All Based on the related works that presented above, there are very few measurements techniques that are focused on the language syntax as a quality factor. In addition, most of the them are using the language standards as a testing method rather than a quality measurement.
In this research, a new process of quality measurement has been introduced based on the programming language standards.
In the proposed model, shown in fig. 1, the piece of code in any programming language, which will be measured, and the standards of the same programming language, which will be used to measure the code, are the two inputs of the proposed measurement model.
After finish reading the code, the model calculates the percentage of satisfied number of rules as in the equation (number of satisfied rules/ (number of satisfied rules + number of violated rules)). Also, for each line that violate a rule, address the violated rules with a suggestion how to improve the code line. At the end of model, a report will be printed that shows the satisfaction percentage, number of violated rules, and a suggestion to improve the code to get higher satisfaction percentage.

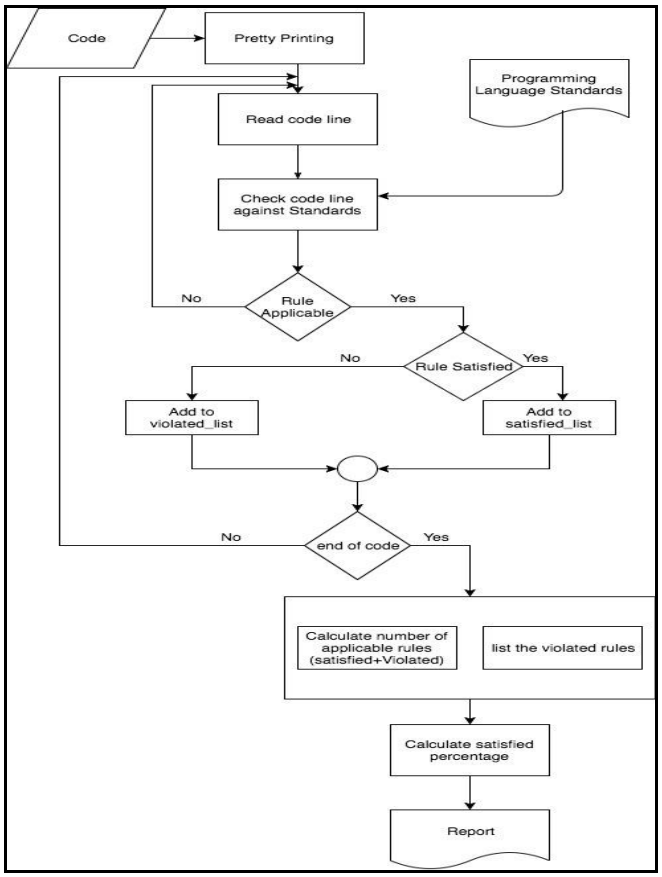

Fig. 1.: The proposed quality measurement framework

4. Evaluation

The proposed framework is measuring the program quality depending on the why it was written, by measuring how many standard rules has followed. The proposed framework is applicable to any programming language and any standards for that language. For example; a program written in Java can be measured against Java language standards such as Google Standards, SUN standards …etc. The proposed framework also can be used to measure the full program or any piece of the code where it does not need to be compiled to be measured. However, in the proposed framework if a code has a syntax error makes it fail to debug but written in a nice way following the language standards, it can get a high satisfaction degree. Therefore, it is recommended to add a condition to the model that the program must be debugged and run successfully.
Another issue that the proposed framework does not address is the weight of satisfied or violated rules. Not all code has the same effect on the rest of the program. There is some piece of codes has a large effect on the program, which means that the applied rules on that piece of code has higher effect and importance than another piece of code that affect nothing but itself. Thus, there must be a way to weight applied rules.
LDRA TBmisra [32], was used to evaluate the program written in C programming language against MISRA C rules. TBmisra uses all the MISRA C rules to measure the code correctness. Even TBmisra shows which rules have been violated; it does not show which of the rules were applied and which were not. It also does not show the percentage of rules satisfaction. Furthermore, the results from TBmisra did not show the effect of these violations on the rest of the program, same as the proposed framework. Therefore, all rules in the LDRA evaluation have the same weight, and the results do not identify the rule that has a major or minor bad effect on program robustness.

FlexeLint [29], is another tool measuring the code using programming language standards. It measures programs written in C using MISRA C rules according to their interpretation. FlexeLint is similar to LDRA TBmisra, where both of them did not show which rules were applied or satisfied and only showed the ones that were violated. FlexeLint only applies static analysis on the assessment since they applied only a static rule to evaluate the program robustness.
For Java programs, there are several measurement tools [33], or to be precise they are analysis tool. PMD is a static Java source code analyser. It uses rule-sets to define when a piece of source is invalid. PMD includes a set of built-in rules and supports the ability to write custom rules. Typically, issues reported by PMD are not true errors, but rather inefficient code, i.e. the application could still function properly even if they were not corrected.

5. Conclusion

Software Measurement is quality attributes that the software product or process should have. It is important since it reduce the efforts and cost of maintenance and evolution. In this research, a framework model is proposed to measure the programs syntax code using the program syntax standards. The framework is programming language independent and all standards of the programming language can be used to measure the code quality.
In this model, code is analyzed and measured against the selected standards. As a result, the code will be giving a mark presents the standards rules satisfaction as percentage. In the future work, the model going to be fully automatic and the standards rules should be weighted. So, the measurement will be depending in the weight not only on the number of satisfied or violated rules.

6. Acknowledgment

This work is part of a funded project by Al-Zaytoonah University of Jordan, under the agreement number 11/10/2016.

References

[1] T. L. J. Ferris, "A new definition of measurement," Measurement, vol. 36, pp. 101-109, 2004/07/01/ 2004.
[2] V. Kharytonov. (2012). Software Measurement: Its Estimation and Metrics Used. Available: http://it-cisq.org/software-meausrement-estimation-metrics/
[3] F. Pinciroli, "Improving Software Applications Quality by Considering the Contribution Relationship Among Quality Attributes," Procedia Computer Science, vol. 83, pp. 970-975, 2016/01/01/ 2016.
[4] K. P. Srinivasan, "Unique Fundamentals of Software Measurment and Software Metrics in Software Engineering " International Journal of Computer Science & Information Technology (IJCSIT), vol. 7, pp. 29-43, 2015.
[5] N. Condori-Fernandez and P. Lago, "Characterizing the Contribution of Quality Requirements to Software Sustainability," Journal of Systems and Software.
[6] W. H. B. W. Hassim, "A Review on Effective Requirement Elicitation Techniques," International Journal of Advances in Computer Science and Technology, vol. 6, pp. 4-8, 2017.
[7] A. S. Guinea, "A Design Methodology for Software Measurement Programs," IEEE Transactions on Software Engineering, 2013.
[8] J. Huang, J. W. Keung, F. Sarro, Y.-F. Li, Y. T. Yu, W. K. Chan, et al., "Cross-validation based K nearest neighbor imputation for software quality datasets: An empirical study," Journal of Systems and Software, vol. 132, pp. 226-252, 2017/10/01/ 2017.
[9] M. Abdallah and M. Al-rifaee, "Java Standards: A Comparative Study," International Journal of Computer Science and Software Engineering (IJCSSE), vol. 6, pp. 146-151, 2017.
[10] A. Ngah, M. Munro, and M. Abdallah, "An Overview of Regression Testing," Journal of Telecommunication, Electronic and Computer Engineering (JTEC), vol. 9, pp. 45-49, 2017.
[11] V. Ivanov, A. Reznik, and G. Succi, "Comparing the reliability of software systems: A case study on mobile operating systems," Information Sciences, vol. 423, pp. 398-411, 2018/01/01/ 2018.
[12] D. Amara and L. B. Arfa Rabai, "Towards a New Framework of Software Reliability Measurement Based on Software Metrics," Procedia Computer Science, vol. 109, pp. 725-730, 2017/01/01/ 2017.
[13] R. M. Carvalho, R. M. d. C. Andrade, and K. M. de Oliveira, "AQUArIUM - A suite of software measures for HCI quality evaluation of ubiquitous mobile applications," Journal of Systems and Software, vol. 136, pp. 101-136, 2018/02/01/ 2018.
[14] Software Metrics - SEI Curriculum Module SEI-CM-12-1.1, 1991.
[15] J. Huang and J. Liu, "A similarity-based modularization quality measure for software module clustering problems," Information Sciences, vol. 342, pp. 96-110, 2016/05/10/ 2016.
[16] G. A. García-Mireles, M. Á. Moraga, F. García, C. Calero, and M. Piattini, "Interactions between environmental sustainability goals and software product quality: A mapping study," Information and Software Technology, 2017/10/09/ 2017.
[17] C. I. M. Bezerra, R. M. C. Andrade, and J. M. Monteiro, "Exploring quality measures for the evaluation of feature models: a case study," Journal of Systems and Software, vol. 131, pp. 366-385, 2017/09/01/ 2017.
[18] A. R. Rezaei, T. Çelik, and Y. Baalousha, "Performance measurement in a quality management system," Scientia Iranica, vol. 18, pp. 742-752, 2011/06/01/ 2011.
[19] D. Pawade, D. J. Dave, and A. Kamath, "Exploring software complexity metric from procedure oriented to object oriented," in 2016 6th International Conference - Cloud System and Big Data Engineering (Confluence), 2016, pp. 630-634.
[20] M. Bundschuh and C. Dekkers, "Object-Oriented Metrics," in The IT Measurement Compendium: Estimating and Benchmarking Success with Functional Size Measurement, ed Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 241-255.
[21] M. Lanza and R. Marinescu, Object-Oriented Metrics in Practice: Springer, 2006.
[22] J. Sterne, Web Metrics: Proven Methods for Measuring Web Site Success: Wiley, 2002.
[23] M.-C. Lee and T. Chang, "Software Measurement and Software Metrics in Software Quality," International Journal of Software Engineering and Its Applications, vol. 7, pp. 15-34, 2013.
[24] M. Abdallah, "A Weighted Grid for Measuring Program Robustness," PhD, Computer Science, Durham University, Durham University, 2012.
[25] M. Abdallah, M. Munro, and K. Gallagher, "Certifying software robustness using program slicing," presented at the IEEE International Conference on Software Maintenance, Timisoara, Romania, 2010.
[26] MISRA-C: 2004, Guidelines for the use of the C language in critical systems, MISRA, 2004.
[27] M. M. A. Abdallah and H. A. Tamimi, "Clauser: Clause Slicing Tool for C Programs," International Journal of Software Engineering and Its Applications, vol. 10, pp. 49-56, 2016.
[28] LDRA, "LDRA Test Suite," ed, 2017.
[29] G. S. LLC, "FlexeLint," 9.00 ed, 2017.
[30] C. Tsalidis, D. Christodoulakis, and D. Maritsas, "Athena: A software measurement and metrics environment," Journal of Software: Evolution and Process, vol. 4, pp. 61-81, 1992.
[31] T. Berardi. (2017, 5/12/2017). Code Quality Standards Highlighted in U.S. State Department CSM (Consular Systems Modernization) Project. Available: http://it-cisq.org/code-quality-standards-highlighted-in-u-s-state-department-csm-consular-systems-modernization-project/
[32] C. Ned and P. D. Susan, "Characteristics of a structured program," vol. 13, ed: ACM, 1978, pp. 36-45.
[33] G. S. LLC, "FlexeLint," vol. 9, ed. http://www.gimpel.com/html/flex.htm, 2016.